Parallel Computing

Portions of this PPT draw from PPTs
by Virendra Singh Yadav

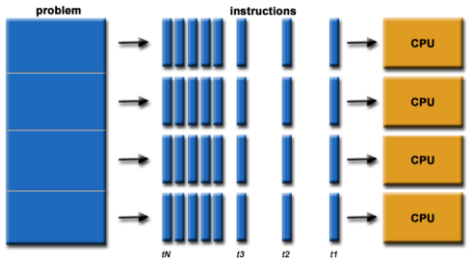
Overview

- What is Parallel Computing
- Why Use Parallel Computing
- Concepts and Terminology
- Parallel Computer Memory Architectures
- Parallel Programming Models
- Examples
- Conclusion

What is Parallel Computing?

- Parallel computing is the simultaneous use of multiple compute resources to solve a computational problem.

An Example



Why Use Parallel

- Saves time
- Solve larger problems
- Cost savings
- Provide concurrency

Types Of Parallelism

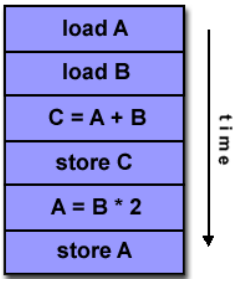
- Data Parallelism
- Task Parallelism

Flynn's Classical Taxonomy

- Distinguishes multi-processor architecture by instruction and data
- SISD – Single Instruction, Single Data
- SIMD – Single Instruction, Multiple Data
- MISD – Multiple Instruction, Single Data
- MIMD – Multiple Instruction, Multiple Data

7

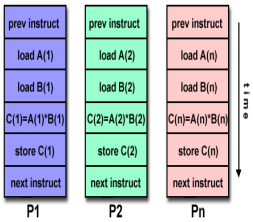
Flynn's Classical Taxonomy:



- Serial
- Only one instruction and data stream is acted on during any one clock cycle

8

Flynn's Classical Taxonomy:



- All processing units execute the same instruction at any given clock cycle.
- Each processing unit operates on a different data element.

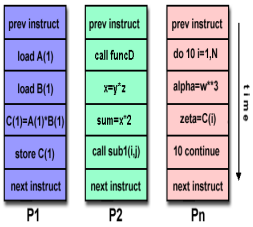
9

MISD

- Different instructions operated on a single data element.
- Example: Multiple cryptography algorithms attempting to crack a single coded message.

10

Flynn's Classical Taxonomy:

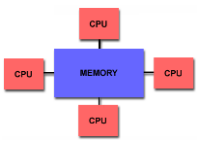


- Can execute different instructions on different data elements.
- Most common type of parallel computer.

11

Parallel Computer Memory

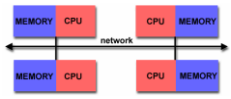
- All processors access all memory as a single global address space.
- Data sharing is fast.
- Lack of scalability between memory and CPUs



12

Parallel Computer Memory

- Each processor has its own memory.
- Is scalable, no overhead for cache coherency.
- Programmer is responsible for many details of communication between processors.



13

Parallel Programming Models

- Shared Memory Model
- Messaging Passing Model
- Data Parallel Model

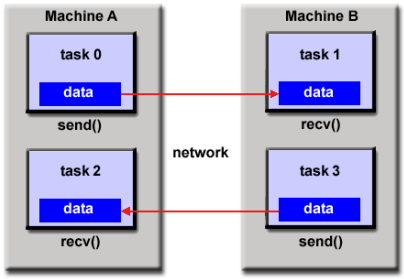
14

Parallel Programming Models: Shared Memory Model

- In the shared-memory programming model, tasks share a common address space, which they read and write asynchronously.
- Locks may be used to control shared memory access.
- Program development can be simplified since there is no need to explicitly specify communication between tasks.

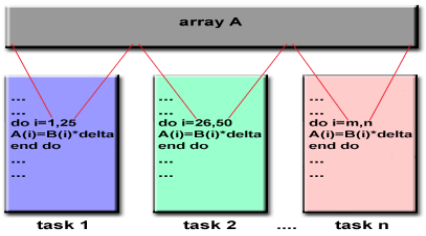
15

Parallel Programming Models: Message Passing Model



16

Parallel Programming Models: Data Parallel Model



17

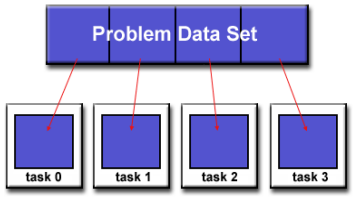
Designing Parallel Programs

- Partitioning
  - Domain Decomposition
  - Functional Decomposition
- Communication
- Synchronization

18

Partition :

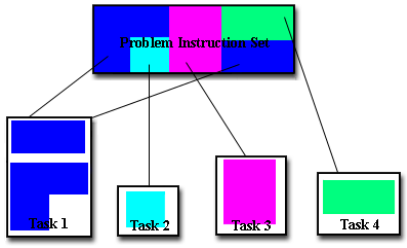
- Each task handles a portion of the data set.



19

Partition

- Each task performs a function of the overall work



20

Designing Parallel Programs Communication

- Synchronous communications are often referred to as *blocking communications since other* work must wait until the communications have completed.
- Asynchronous communications allow tasks to transfer data independently from one another.

21

Designing Parallel Programs Synchronization

Types of Synchronization:

- Barrier
  - Each task performs its work until it reaches the barrier
  - When the last task reaches the barrier, all tasks are synchronized
- Lock / semaphore
  - Typically used to protect access to global data or code section
  - Only one task at a time may use the lock
- Synchronous communication operations

22

Example:

- As a simple example, if we are running code on a 2-processor system (CPUs "a" & "b") in a parallel environment and we wish to do tasks "A" and "B" , it is possible to tell CPU "a" to do task "A" and CPU "b" to do task 'B" simultaneously, thereby reducing the runtime of the execution.

23

Example: Array Processing

- Serial Solution
  - Perform a function on a 2D array.
  - Single processor iterates through each element in the array
- Possible Parallel Solution
  - Assign each processor a partition of the array.
  - Each process iterates through its own partition.

24

Conclusion

- Parallel computing is fast.
- There are many different approaches and models of parallel computing.

25

Problems to Consider

- Data dependency
- Load balancing

26

References

- https://computing.llnl.gov/tutorials/parallel_comp
- Introduction to Parallel Computing, www.llnl.gov/computing/tutorials/parallel_comp/#Whatis
- www.cs.berkeley.edu/~yelick/cs267-sp04/lectures/01/lect01-intro
- http://www-users.cs.umn.edu/~karypis/parbook/

27